A Writing an ASI File

How can I create my own ASI file from scratch or modify an existing one?

The ASI file format is used to store data, in a hierarchy of a scenario, its platforms, beams, and gates. As the ASI file format has evolved, features such as Generic Data, described later in this section, have been added for the sake of convenience. These features should be used at the user's discretion; an ASI file should be an accurate representation of the actual data for a scenario. Although these features can be used to simulate "scripted" events over time, we recommend against using generic data in this manner. Since generic data is considered data, it takes precedence over similar user preference settings and can result in undesired display behavior. If you desire to change the way actual data is presented, we recommend using the SIMDIS Python scripting interface.

The ASI format is used by SIMDIS to generate the binary `.fct` files. The program utilizes a file parser to match keywords with the incoming data. The file parser recognizes white space, e.g. tabs or spaces, as delimiters. The file parser also skips all blank lines and lines beginning with shell script (#) or C++ style (//) comments. For example, the following lines would be skipped in the parser.

```
#   This is skipped
//  This is skipped
```

Furthermore, quoted strings must contain a value. The parser will register any empty quoted string, “”, as an error.

Throughout this section, [**REQUIRED**] fields will be bold face and color-coded red, while [**OPTIONAL**] fields will be bold face and color-coded teal. Special note sections will be marked in bold face type. Pseudo code and file format examples will be displayed using a fixed width font.

A.1 File Initialization

The ASI file initialization phase is comprised of two [**REQUIRED**] sections and two [**OPTIONAL**] sections.

Scenario Initialization	**[REQUIRED]**
Platform Initialization	**[REQUIRED]**
Beam Initialization	**[OPTIONAL]**
Gate Initialization	**[OPTIONAL]**

A.1.1 Scenario Initialization

There are three keywords that are [**REQUIRED**] to be specified during the scenario initialization phase. These are as follows:

```
Version       16
RefLLA        22.0 -160.0 0.0
CoordSystem   "LLA"
```

The < **Version** > keyword enables multiple versions of the file format to be maintained.

NOTES:

Version 1 is no longer supported. Versions 2 through 16 are supported.

For SIMDIS version 9.1.0, the supported ASI file version number is 16.

The < **RefLLA** > keyword specifies the reference latitude, longitude and altitude of the flat earth or tangent plane coordinate system origin for the scenario. The reference altitude value will not affect the height values for data entered as LLA (Geodetic) or one for the Earth Centered systems; it only affects heights for flat earth or tangent plane systems. The input units are always degrees for latitude and longitude and meters for altitude. Negative latitude values are south of the Equator, positive values are north. Negative longitude values are west of the Prime Meridian, positive values are east.

NOTES:

The **RefLat**, **RefLon** and **RefAlt** keywords have been deprecated as of ASI version 15. The commands have been replaced with < **RefLLA** >.

Latitude and longitude can be specified in degrees decimal (DD), degrees minutes (DM) or degrees minutes seconds (DMS). For both the DM and DMS notations, the values must be enclosed within quotes in order to be interpreted as a single value. Examples are provided below:

```
// degrees minutes decimal (DM) notation
RefLLA  "21 34.88243" "-159 12.47353" 0.0

// degrees minutes seconds (DMS) notation
RefLLA  "21 34 52.946" "-159 12 28.412" 0.0
```

The < **CoordSystem** > keyword specifies the type of coordinate system the incoming data is represented in. Supported formats are:

Value	Meaning
NED	NED Scaled Flat Earth
NWU	NWU Scaled Flat Earth
ENU	ENU Scaled Flat Earth
LLA	Latitude, Longitude, Altitude (Geodetic)
ECEF	Earth Centered Earth Fixed, WGS-84 Ellipsoid
ECI	Earth Centered Inertial, WGS-84 Ellipsoid
XEAST	X-East Tangent Plane
GTP	Generic Tangent Plane

The Scenario Initialization phase also supports 18 [**OPTIONAL**] keywords. They are:

```
ScenarioInfo        "Data from various sources"
WindAngle           130.4172209
WindSpeed           1.313837
ReferenceYear       2006
Classification      "UNCLASSIFIED" 0x8000FF00
DEDMap              "Kauai"
WVSMap              "Kauai1.wvs"
GOGFile             "us_states.gog"
SoundFile           "file.audio" 234.0 455.0
ViewFile            "19991109.view"
MediaFile           "missile.lst"
ITConfigFile        "configPMRF.txt"
RuleFile            "simdis_gps.rul"
DegreeAngles        1
MagneticVariance    "TRUE"
VerticalDatum       "WGS84"
ReferenceTimeECI    "0.0"
TangentPlaneOffset -14063.024 5641.235 13.145999
```

The < **ScenarioInfo** > keyword specifies a text based description of the scenario. The input format is a character string.

The < **WindAngle** > keyword specifies the inertial direction, angle off of North, a.k.a. true, the wind is coming from. A WindAngle of 90.0 specifies the wind coming from the east heading west. The input units are always degrees.

The < **WindSpeed** > keyword specifies the velocity of the wind. The input units are meters per second.

The < **ReferenceYear** > keyword specifies the year that time is referenced to. All time values are assumed to be seconds since the beginning of a year.

The < **Classification** > keyword specifies the classification label of data contained in the file, in addition to a color (color formats are described later in this section) for the classification label. The default setting is:

"UNCLASSIFIED" 0x8000FF00

The < **DEDMap** > keyword specifies a DED (Digital Elevation Data) map to load and display in SIMDIS. See `$(SIMDIS_DIR)`/data/DED for available maps. Additionally, more than one map can be loaded using a single keyword. For example:

DEDMAP " ChinaLake "

DEDMAP "ChinaLake" "California_South"

The < **WVSMap** > keyword specifies a WVS (World Vector Shoreline) map to load and display in SIMDIS. See either `$(SIMDIS_DIR)`/data/WVS/maps or `$(SIMDIS_DIR)`/data/WVS/lmaps for available maps. Additionally, more than one map can be loaded using a single keyword. For example:

WVSMap "Kauai1.wvs"

WVSMap "Kauai1.wvs" "world40.lwvs"

The < **GOGFile** > keyword specifies a GOG file to load and display in SIMDIS. See `$(SIMDIS_DIR)`/data/GOG for available files. Additionally, more than one file can be loaded using a single keyword. For example:

GOGFile "us_states.gog"

GOGFile "bsure.gog" "swtr.gog"

The < **SoundFile** > keyword specifies a sound file to load and play in SIMDIS. Start and end times associated to the data are also required. Refer to the section below on Time Formats for more information. Only one sound file per data file is permitted.

The < **ViewFile** > keyword specifies a view (eye position) file to load in SIMDIS. Only one view file per data file is permitted. The view file is generated from within SIMDIS. Use the View Dialog (Alt V) to load, create, and save eye positions into a .view file.

The < **MediaFile** > keyword specifies a SIMDIS media player file (.tmd, .lst, .pst) to load in SIMDIS. Only one media file per data file is permitted.

The < **ITConfigFile** > keyword specifies an imagery and terrain configuration file to load in SIMDIS. Only one imagery and terrain configuration file per data file is permitted. The imagery and terrain configuration file is generated from within SIMDIS. Use the Terrain Tool Dialog (Alt I) to load, create, and save imagery and terrain configurations into a text (.txt) file.

The < **RuleFile** > keyword specifies a preference rule file to load in SIMDIS. Only one preference rule file per data file is permitted. The preference rule file is generated from within SIMDIS. Use the Preference Rules Dialog (Alt P) to load, create, and save rules into a .rul file.

The < **DegreeAngles** > keyword specifies whether or not Platform, Beam and Gate data angles should be in degrees or radians.

Value	Meaning
0	Angle units interpreted as radians [**Default**]
1	Angle units interpreted as degrees

NOTE: The DegreeAngles setting does not apply to the latitude and longitude values of RefLLA.

The < **MagneticVariance** > keyword specifies the magnetic variance (declination) between magnetic north and true north. The variance is considered positive east of true north and negative when west. SIMDIS supports three options for specifying magnetic variance data:

Value	Meaning
WMM	Variance is computed based on the World Magnetic Model.
TRUE	No variance, yaw values relative to true north. [**Default**]
14.2	User specified variance in DD, DM or DMS.

If a magnetic variance is not specified, then all yaw values will be referenced to true north. WMM data is only valid for a five-year period. Presently, SIMDIS can predict the magnetic variance using the WMM for data within the ranges of 1985 to 2010. Additional information regarding the WMM can be found on the National Geophysical Data Center's (NGDC) web site: http://www.ngdc.noaa.gov/seg/WMM/back.shtml. For a user specified variance, the magnetic declination is considered as the angle between the local magnetic field, the direction the north end of a compass points and true north. The declination is positive when the magnetic north is east of true north. Units for the user specified variance must be in degrees. The table below summarizes the availability of the magnetic variance per coordinate system:

Coordinate System	WMM	TRUE	User
Tangent Plane	Yes	Yes	Yes
Scaled Flat Earth	Yes	Yes	Yes
LLA (Geodetic)	Yes	Yes	Yes
Earth Centered	N/A	N/A	N/A

The < **VerticalDatum** > keyword specifies the zero surface to which elevations or heights are referenced. This setting will be ignored if the input data is in one of the Earth Centered coordinate systems. SIMDIS supports three options for specifying vertical datum values:

Value	Meaning
WGS84	Surface referenced to the WGS-84 ellipsoid. [**Default**]
MSL	Mean sea level surface based on the EGM96 model.
7.4	User specified vertical datum, in meters.

The WGS84 setting is equivalent to no vertical datum being specified, as all height values within SIMDIS are referenced to the WGS-84 ellipsoid. In order to approximate Mean Sea Level, SIMDIS uses the WGS-84 Earth Gravitational Model 1996 (EGM96). This model consists of a height file with a 0.25 degree grid of point values in the tide-free system. Bi-linear interpolation is used to obtain the associated height value based on a specified latitude and longitude. Additional information on the EGM96 can be found on the National Geospatial-Intelligence Agency's web site: http://earth-info.nga.mil/GandG/wgs84/gravitymod/index.html.

The table below summarizes the availability and reference surface of the vertical datum offsets per coordinate system:

Coordinate System	WGS84 (none)	MSL	User
Tangent Plane	No offsets applied	N/A	Yes, datum offset from the reference altitude setting
Scaled Flat Earth	No offsets applied	N/A	Yes, datum offset from the reference altitude setting
LLA (Geodetic)	No offsets applied	Yes, datum offset from EGM geoid	Yes, datum offset from the surface of the ellipsoid
Earth Centered	N/A	N/A	N/A

The < **ReferenceTimeECI** > keyword specifies the time at which the Earth Centered Inertial reference frame is defined. Refer to the section below on Time Formats for more information. If this value is not specified, the time of the first data point for each individual platform will be used as that platform's reference time. SIMDIS uses the ECI reference time to calculate an elapsed time which is multiplied by the earth's angular velocity in order to determine the corresponding ECEF position.

The < **TangentPlaneOffset** > keyword specifies the x and y translation positions and a rotation angle to offset the Generic Tangent Plane (GTP) coordinate system. The x offset position is the true east distance; the y offset position is the true north distance in meters of the desired origin as seen from the tangential point. The x offset is positive if the origin lies to the east of the tangential point, and it is negative if the origin lies to the west. The y offset is positive if the origin lies to the north of the tangential point, and it is negative if the origin lies to the south. The rotation angle is the desired angle in degrees to apply to rotate the x-y plane. Positive angles perform a clockwise rotation referenced to true north. If the tangent plane offset values are not set, the translation and rotation values default to zero. When the translation and rotation values are zero, the GTP coordinate system becomes an X-East tangent plane.

A.1.2 Platform Initialization

There are three keywords that are [**REQUIRED**] to be specified during the platform initialization phase. These keywords are repeated based on how many unique platforms are in the scenario. Using two unique platforms, the keywords are as follows:

```
PlatformID      1
PlatformName    1      "CG-70 Lake Erie"
PlatformIcon    1      "uss_ticonderoga_class_cg"

PlatformID      2
PlatformName    2      "bogey"
PlatformIcon    2      "bqm-74e"
```

The < **PlatformID** > keyword specifies a unique id that is assigned to this platform. This id is used to associate data to this platform during the importing phase. The value must be a positive non-zero integer.

The < **PlatformName** > keyword specifies the unique platform id and a description, i.e. a radio call sign that is assigned to this platform. The input units are a character string. For strings separated by spaces, use double quotes, i.e.

```
PlatformName    1      "DDG-69 Milius"
```

The < **PlatformIcon** > keyword specifies the unique platform id and a 3-D model file that is used to depict the platform. See `$(SIMDIS_DIR)`/data/models for available files. Values are strings and the file extension is optional, i.e.

```
PlatformIcon    1      "uss_arliegh_burke_class_ddg"
```

The Platform Initialization phase also supports 14 [**OPTIONAL**] keywords. The keywords are as follows:

```
PlatformType                2          "ship"
PlatformFHN                 2          "F"
PlatformOffset              2          0.0 0.0 1.5
PlatformOriOffset           2          90.0 0.0 0.0
PlatformRCS                 2          "fake_rcs_3.rcs"
PlatformAttachedGOG         2          "t2.rxy"
PlatformInterpolate         2          1
OriginalID                  2          14
PlatformRefLLA              2          22. -160. 0.
PlatformCoordSystem         2          "ECEF"
PlatformMagneticVariance    2          "TRUE"
PlatformVerticalDatum       2          "WGS84"
PlatformReferenceTimeECI    2          "0.0"
PlatformTangentPlaneOffset  2          -14063.024 5641.235 13.145999
PlatformUsesQuaternion      2          0
```

The < **PlatformType** > keyword specifies the unique platform id and a unique category that the platform is a member of. The available categories are:

Value	Meaning
0 or "unknown"	Unknown
1 or "ship"	Ship
2 or "submarine"	Submarine
3 or "aircraft"	Aircraft
4 or "satellite"	Satellite
5 or "helicopter"	Helicopter
6 or "missile"	Missile
7 or "decoy"	Decoy
8 or "buoy"	Buoy
9 or "reference"	Reference
10 or "vehicle"	Vehicle (cars, trucks, tanks, etc)
11 or "landsite"	Land Site (buildings, airfields, radar installations, etc)
12 or "torpedo"	Torpedo
13 or "contact"	Contact (radar, sonar, visual, etc)

The input units are either integer or strings. In reality this keyword will create a category data value based on one of the standard SIMDIS platform type values. The following commands are equivalent:

```
PlatformType  2  "ship"
CategoryData  2  -1     "Platform Type"   "ship"
```

The < **PlatformFHN** > keyword specifies the platform unique id and the platform's friendly, hostile or neutral designation. Valid values are:

Value	Meaning
F	Friendly
H	Hostile
N	Neutral

If no value is set, then the platform defaults to a friendly designation. This keyword is actually a macro to create category data. The following commands are equivalent:

```
PlatformFHN    2   "F"
CategoryData   2   -1   "Affinity"   "Friendly"
```

The < **PlatformOffset** > keyword specifies the platform unique id and the platform's XYZ body offset with respect to the platform center. The input units are meters.

The < **PlatformOriOffset** > keyword specifies the platform unique id and the platform's orientation offset with respect to the platform body. Orientations are specified in the following order, course about the Z axis; pitch about the Y axis and roll about the X axis. The input units are degrees.

The < **PlatformRCS** > keyword specifies the platform unique id and the radar cross-section (RCS) filename. RCS file formats are determined during loading. Currently, the following file formats are supported:

Value	Meaning
LUT	(ASCII lookup table file format, see later section for more details)
SADM	(Ship Air Defence Model file format, see later section for more details)

The < **PlatformAttachedGOG** > keyword specifies the platform unique id and the filename of a relative GOG pattern. See `$(SIMDIS_DIR)`/data/GOG for available files.

The < **PlatformInterpolate** > keyword specifies the platform unique id and a value 0 or 1. The value indicates whether position, orientation and velocity data is interpolated between discrete data points during display in SIMDIS.

Value	Meaning
0	Interpolation between data points is not performed
1	Interpolation between data points is performed [**Default**]

The **PlatformInterpolate** keyword replaces the **PlatformInterpolateOri** as interpolation is now controlled for all data, not just orientation. If the **PlatformInterpolateOri** keyword is encountered in older ASI files it will now behave as if it were the newer **PlatformInterpolate** keyword.

The < **OriginalID** > keyword specifies the platform unique id and an ID that can be used to map an entity to an original data source. This does not have to be unique. SIMDIS plug-ins will be able to access the OriginalID value, but will not be able to access an entity's UniqueID as set in the ASI file. Additionally, the UniqueID might change as a file is converted from ASI to FCT and back to ASI again, but the OriginalID field will stay the same.

The < **PlatformRefLLA** > keyword specifies the reference latitude, longitude and altitude of the flat earth or tangent plane coordinate system origin for the platform. The reference altitude value will not affect the height values for data entered as LLA (Geodetic) or one for the Earth Centered systems; it only affects heights for flat earth or tangent plane systems. The input units are always degrees for latitude and longitude and meters for altitude. Negative latitude values are south of the Equator, positive values are north. Negative longitude values are west of the Prime Meridian, positive values are east. Latitude and longitude values can be input using DD, DM or DMS notation. If this value is not specified, the scenario's reference LLA values are used.

The < **PlatformCoordSystem** > keyword specifies the platform unique id and the type of coordinate system the incoming platform's data is represented in. Supported formats are:

Value	Meaning
NED	NED Scaled Flat Earth
NWU	NWU Scaled Flat Earth
ENU	ENU Scaled Flat Earth
LLA	Latitude, Longitude, Altitude (Geodetic)
ECEF	Earth Centered Earth Fixed, WGS-84 Ellipsoid
ECI	Earth Centered Inertial, WGS-84 Ellipsoid
XEAST	X-East Tangent Plane
GTP	Generic Tangent Plane

If this value is not specified, the scenario's coordinate system value is used.

The < **PlatformMagneticVariance** > keyword specifies the platform unique id and the magnetic variance (declination) for this platform between magnetic north and true north. The variance is considered positive east of true north and negative when west. SIMDIS supports three options for specifying magnetic variance data:

Value	Meaning
WMM	Variance is computed based on the World Magnetic Model.
TRUE	No variance, yaw values relative to true north. [**Default**]
14.2	User specified variance in DD, DM or DMS.

WMM data is only valid for a five-year period. Presently, SIMDIS can predict the magnetic variance using the WMM for data within the ranges of 1985 to 2010. Additional information regarding the WMM can be found on the National Geophysical Data Center's (NGDC) web site: http://www.ngdc.noaa.gov/seg/WMM/back.shtml. For a user specified variance, the magnetic declination is considered as the angle between the local magnetic field, the direction the north end of a compass points and true north. The declination is positive when the magnetic north is east of true north. Units for the user specified variance must be in degrees. If this value is not specified, the scenario's magnetic variance value is used. If a magnetic variance is not specified by either the scenario or the platform, then all yaw values will be referenced to true north. This setting will be ignored if the input data is in one of the Earth Centered coordinate systems. The table below summarizes the availability of the magnetic variance per coordinate system:

Coordinate System	WMM	TRUE	User
Tangent Plane	Yes	Yes	Yes
Scaled Flat Earth	Yes	Yes	Yes
LLA (Geodetic)	Yes	Yes	Yes
Earth Centered	N/A	N/A	N/A

The < **PlatformVerticalDatum** > keyword specifies the platform unique id and the zero surface to which elevations or heights for this platform are referenced. SIMDIS supports three options for specifying vertical datum values:

Value	Meaning
WGS84	Surface referenced to the WGS-84 ellipsoid. [**Default**]
MSL	Mean sea level surface based on the EGM96 model.
7.4	User specified vertical datum, in meters.

In order to approximate Mean Sea Level, SIMDIS uses the WGS-84 Earth Gravitational Model 1996 (EGM96). This model consists of a height file with a 0.25 degree grid of point values in the tide-free system. Bi-linear interpolation is used to obtain the associated height value based on a specified latitude and longitude. Additional information on the EGM96 can be found on the National Geospatial-Intelligence Agency's web site: http://earth-info.nga.mil/GandG/wgs84/gravitymod/index.html. If this value is not specified, the scenario's vertical datum value is used. If a vertical datum is not specified by either the scenario or the platform, then all height values will be referenced to the WGS-84 ellipsoid. The table below summarizes the availability and reference surface of the vertical datum offsets per coordinate system:

Coordinate System	WGS84 (none)	MSL	User
Tangent Plane	No offsets applied	N/A	Yes, datum offset from the reference altitude setting
Scaled Flat Earth	No offsets applied	N/A	Yes, datum offset from the reference altitude setting
LLA (Geodetic)	No offsets applied	Yes, datum offset from EGM geoid	Yes, datum offset from the surface of the ellipsoid
Earth Centered	N/A	N/A	N/A

The < **PlatformReferenceTimeECI** > keyword specifies the platform unique id and the time at which the Earth Centered Inertial reference frame for this platform is defined. Refer to the section below on Time Formats for more information. If this value is not specified, the scenario's reference time value is used. If the neither the scenario nor the platform's reference ECI time is specified, the time of the first data point will be used as the reference time. SIMDIS uses the ECI reference time to calculate an elapsed time which is multiplied by the earth's angular velocity in order to determine the corresponding ECEF position.

The < **PlatformTangentPlaneOffset** > keyword specifies the platform unique id and the x and y translation positions and a rotation angle to offset the Generic Tangent Plane (GTP) coordinate system for this platform. The x offset position is the true east distance; the y offset position is the true north distance in meters of the desired origin as seen from the tangential point. The x offset is positive if the origin lies to the east of the tangential point, and it is negative if the origin lies to the west. The y offset is positive if the origin lies to the north of the tangential point, and it is negative if the origin lies to the south. The rotation angle is the desired angle in degrees to apply to rotate the x-y plane. Positive angles perform a clockwise rotation referenced to true north. If this value is not specified, the scenario's tangent plane offset values are used.

The < **PlatformUsesQuaternion** > keyword specifies the platform unique id and how to interpret orientation data input as a quaternion scalar and vector value. The third argument indicates the position of the quaternion scalar value with respect to the quaternion vector. SIMDIS supports four positions of the quaternion scalar:

Value	Meaning
0	Quaternion in the form of q0 + q1i + q2j + q3k [**Default**]
1	Quaternion in the form of q3k + q0 + q1i + q2j
2	Quaternion in the form of q2j + q3k + q0 + q1i
3	Quaternion in the form of q1i + q2j + q3k + q0

A.1.3 Beam Initialization

If < **BeamID** > keywords are found within the .asi file this section is [**REQUIRED**]. Otherwise it along with the gate initialization sections can be skipped.

There are three keywords that are [**REQUIRED**] to be specified during the beam initialization phase. These keywords are repeated based on the number of unique beams found in the .asi file. Using an example of two unique beams is presented below:

```
BeamID    1      101
VertBW    101    5.0
HorzBW    101    5.0

BeamID    2      102
VertBW    102    8.0
HorzBW    102    8.0
```

The < **BeamID** > keyword specifies the platform unique id to which it is attached, and its own unique id which it is assigned. This id is used to associate data with this beam during the importing phase. The value must be a positive non-zero integer.

The < **VertBW** > keyword specifies the beam unique id and the beam's vertical beam width. The input units are always degrees.

The < **HorzBW** > keyword specifies the beam unique id and the beam's horizontal beam width. The input units are always degrees.

The Beam Initialization phase also supports 14 [**OPTIONAL**] keywords. These are as follows:

```
BeamDesc            102  "Seeker"
AzimOffset          102  5.0
ElevOffset          102  5.0
BodyOffset          102  0.0 0.0 -10.0
BeamMissileOffset   102
AREPSPattern        102  "file_APM_0_15_15.txt"
AntennaPattern      102  "table"     "bogey_ant.aptf"
AntennaAlgorithm    102  "pedestal"
AntennaGain         102  30.0
AntennaPeakPower    102  20000.0
AntennaFrequency    102  1000
AntennaPolarity     102  "UNKNOWN"
BeamInterpolate     102  1
OriginalID          102  114
```

The < **BeamDesc** > keyword specifies the beam unique id and a brief description of the beam.

The < **AzimOffset** > keyword specifies the beam unique id and the beam's azimuth angle offset with respect to the platform. The input units are always degrees.

The < **ElevOffset** > keyword specifies the beam unique id and the beam's elevation angle offset with respect to the platform. The input units are always degrees.

The < **BodyOffset** > keyword specifies the beam unique id and the beam's XYZ body offset with respect to the platform center. The input units are meters.

The < **BeamMissileOffset** > keyword specifies the beam unique id. In ASI versions 2 through 8, a beam would automatically be offset by SIMDIS when the host platform was a missile. This automatic offset would enable the beam to emit from the nose instead of the center of the platform. This functionality is supported for old versions for backwards compatibility. As of ASI version 9, this keyword allows SIMDIS to automatically adjust the beam position to the "nose" of the host platform, regardless of type.

The < **AREPSPattern** > keyword specifies the beam unique id and either the ASCII Advanced Refractive Effects Prediction System (AREPS) pattern (.txt) file or the Advanced Propagation Model (APM) input (.in) file to load. SIMDIS supports the loading of AREPS ASCII file version 3.x.

NOTES:

> More than one AREPS file can be loaded. For additional information regarding AREPS and APM, see the Space and Naval Warfare Systems Center Atmospheric Propagation Branch's web site: http://sunspot.spawar.navy.mil/2858
>
> In order to display either the AREPS or APM data, the RF Propagation Tool is required.

Antenna patterns are available in two flavors, tabular data file or algorithm.

The < **AntennaPattern** > keyword specifies the beam unique id and the beam's pattern type and pattern file. The available pattern types are:

Value	Meaning
TABLE	ASCII file format, see Table Antenna Pattern section for details
RELATIVE	ASCII file format, see Relative Table Antenna Pattern for details
MONOPULSE	ASCII file format, see Monopulse Antenna Pattern for details
BILINEAR	ASCII file format, see Bilinear Antenna Pattern for details
NSMA	ASCII file format, see NSMA Antenna Pattern for details

The < **AntennaAlgorithm** > keyword specifies the beam unique id and the beam's pattern algorithm. The available pattern algorithms are:

Value	Meaning
PEDESTAL	Pedestal antenna pattern model
GAUSS	Gaussian antenna pattern model
CSCSQ	Cosecant squared antenna pattern model
SINXX	Sin (x)/x antenna pattern model
OMNI	Omni directional antenna pattern model

NOTE:

> Both < **AntennaPattern** > and < **AntennaAlgorithm** > keywords are mutually exclusive. You should only use one or the other per Beam.

If either < **AntennaPattern** > or < **AntennaAlgorithm** > is specified, then the following three keywords will be required in order to correctly display the antenna pattern.

The < **AntennaGain** > keyword specifies the beam unique id and the beam's main lobe antenna gain with respect to the antenna pattern. The input units are dB.

The < **AntennaPeakPower** > keyword specifies the beam unique id and the beam's peak power output. The input units are Watts.

The < **AntennaFrequency** > keyword specifies the beam unique id and the beam's transmit frequency. The input units are MHz.

The < **AntennaPolarity** > keyword specifies the beam unique id and the beam's polarization. The available polarization values are:

Value	Meaning
Unknown	Unknown polarization
Horizontal	Horizontal polarization
Vertical	Vertical polarization
Circular	Circular polarization
HorzVert	Horizontal transmit, Vertical receive polarization
VertHorz	Vertical transmit, Horizontal receive polarization
LeftCirc	Left-hand circular polarization
RightCirc	Right-hand circular polarization
Linear	Linear polarization

The < **BeamInterpolate** > keyword specifies whether or not interpolation should be performed between discrete beam data points.

Value	Meaning
0	Interpolation between data points is not performed
1	Interpolation between data points is performed [**Default**]

The **BeamInterpolate** keyword replaces the **BeamInterpolatePos** for consistency with the Platform keyword version. If the **BeamInterpolatePos** keyword is encountered in older ASI files it will now behave as if it were the newer **BeamInterpolate** keyword.

The < **OriginalID** > keyword specifies an ID that can be used to map an object to an original data source. This does not have to be unique. SIMDIS plug-ins will be able to access the OriginalID value, but will not necessarily be able to access an object's UniqueID as set in the ASI file. Additionally, the UniqueID might change as a file is

converted from ASI to FCT and back to ASI again, but the OriginalID field will stay the same.

A.1.4 Gate Initialization

If any < **GateID** > keywords are found within the .asi file this section is [**REQUIRED**]. Otherwise it is skipped.

There is a single keyword that is [**REQUIRED**] to be specified during the gate initialization phase. This keyword is repeated based on each unique gate found in the .asi file. An example using three unique gates is shown below:

```
GateID    101    200
GateID    102    201
GateID    102    202
```

The < **GateID** > keyword specifies the beam unique id to which it is a part of, and its own unique id that is assigned to this gate. This id is used to associate data with this gate during importing phase. The value must be a positive non-zero integer.

The Gate Initialization phase also supports 4 [**OPTIONAL**] keywords. They are as follows:

```
GateDesc              200       "Range Gate"
GateInterpolatePos    200       123
GateType              200       "SECTOR"
OriginalID            200       123
```

The < **GateDesc** > keyword specifies the gate unique id and a brief description of the gate.

The < **GateInterpolate** > keyword specifies whether or not interpolation should be performed between discrete gate data points.

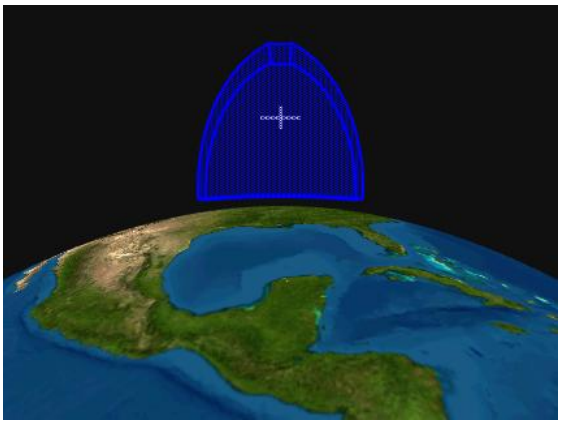
Value	Meaning
0	Interpolation between data points is not performed
1	Interpolation between data points is performed [**Default**]

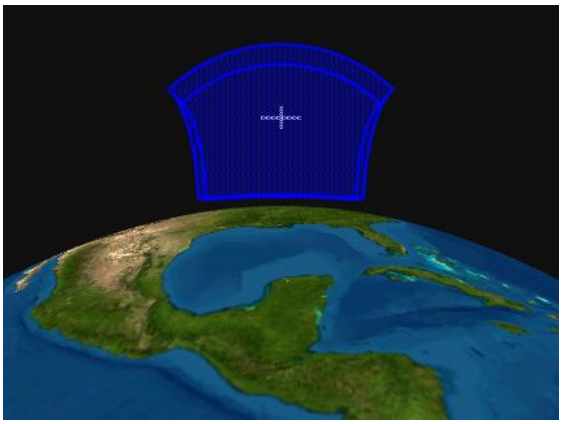
The **GateInterpolate** keyword replaces the **GateInterpolatePos** for consistency with the Platform and Beam keyword versions. If the **GateInterpolatePos** keyword is encountered in older ASI files it will now behave as if it were the newer **GateInterpolate** keyword.

The < **GateType** > keyword specifies the gate unique id and the type of gate to be drawn. Valid values are:

UNKNOWN, RANGE, GUARD, TARGET, ANGLE, RAIN, CLUTTER, FOOTPRINT, PUSH, SECTOR, COVERAGE, and BODY

A "COVERAGE" type gate is drawn as a spherical wedge; the combination of vertical beam width and the elevation pointing value cannot exceed 90 degrees. The remaining gate types are drawn as a spherical patch that can be elevated to 90 degrees. If no value is set, then the gate defaults to an "UNKNOWN" type. The images below depict a “COVERAGE” gate type on the left and the remaining gate types on the right.





NOTE: Coverage and Search Volumes would typically be drawn with a “COVERAGE” gate type while gates for tracking radars would be represented using the other gate types.

The < **OriginalID** > keyword specifies an ID that can be used to map an object to an original data source. This does not have to be unique. SIMDIS plugins will be able to access the OriginalID value, but will not be able to access an object’s UniqueID as set in the ASI file. Additionally, the UniqueID might change as a file is converted from ASI to binary FCT and back to ASI again, but the OriginalID field will stay the same.

All PlatformID, BeamID and GateID values are UNIQUE within an ASI file. It is good practice to use numbers that are spaced far apart to prevent using the unique id more than once. In our example, we use numbers that are spaced by 100. As mentioned above, the unique ID values used to identify a platform might change as the file is converted into FCT and back out to ASI.

A.2 Input Data

Once all initialization sections have been read, the data input can begin. The Input data can be sub-divided into 4 sections, of which 1 is required and 3 are optional.

```
Platform Input Data      [REQUIRED]
Beam Input Data          [OPTIONAL]
Gate Input Data          [OPTIONAL]
Generic Input Data       [OPTIONAL]
```

A.2.1 Platform Input Data

There is a single keyword that is [**REQUIRED**] to be specified during the platform data input phase. The keyword is as follows:

```
// Minimum input, time and position
PlatformData 1  0.0 1115.0 342.0 0.0
// Time, position and orientation
PlatformData 2  0.0 1115.0 342.0 0.0 3.344 0.0 0.0
// Time, position, orientation and speed
PlatformData 3  0.0 1115.0 342.0 0.0 3.344 0.0 0.0 10.0
// Time, position, orientation and velocity vector
PlatformData 4  0.0 1115.0 342.0 0.0 3.344 0.0 0.0 3.0 5.0 2.0
```

Where the < **PlatformData** > keyword specifies the following required fields:

Platform unique id to which it belongs
Time for which this data is valid
X or latitude position of platform (meters, radians or degrees)
Y or longitude position of platform (meters, radians or degrees)
Z position of platform (meters)

And the following Optional fields:

Yaw or Psi orientation of platform (radians or degrees)
Pitch or Theta orientation of platform (radians or degrees)
Roll or Phi orientation of platform (radians or degrees)
Quaternion scalar and vector (q0, q1, q2, q3) if < **PlatformUsesQuaternion** > is set
Speed or Velocity vector (Vx ,Vy,Vz) of platform (meters/sec)

NOTES:

See the Time Formats section for more information on time data formats.

When PlatformData contains only position data, orientation will be generated based on the PlatformType setting of < **CategoryData** >. The following are the default calculations used:

Value	Meaning
"unknown"	Course only
"ship"	Course only
"submarine"	Course only
"aircraft"	Course, pitch and roll
"satellite"	Course and pitch
"helicopter"	Course and pitch
"missile"	Course and pitch
"decoy"	Course and pitch
"buoy"	Course only
"reference"	No orientation
"vehicle"	Course and pitch
"landsite"	No orientation
"torpedo"	Course and pitch
"contact"	No orientation

All angle units are radians unless the scenario keyword < **DegreeAngles** > is set.

If either an ECEF or ECI coordinate system is used, orientation is represented by the Euler angles psi (ψ), theta (θ) and phi (ϕ). The orientation representation of the Euler angles is presented below.

If the keyword < **PlatformUsesQuaternion** > is specified, orientation angles are replaced by a quaternion scalar and vector.

If neither a speed nor a velocity vector is entered, speed will be calculated based on time and distance traveled between two points.

If only speed is specified, a velocity vector will be created based on orientation.

SIMDIS supports three scaled flat earth systems, ENU, NED and NWU. In a Scaled Flat Earth system, the earth's surface is projected (warped) onto an X-Y plane of a Cartesian coordinate system, with the reference origin specified at a latitude and longitude surface location.

The ENU system is represented as:

Positive X axis points East
Positive Y axis points North
Positive Z axis points Up (positive altitude is positive Z)
True North [000] is looking down the Y-axis
East [090] is looking down the X-axis
Positive yaw/course is clockwise from North

The NED system is represented as:

Positive X axis points North
Positive Y axis points East
Positive Z axis points Down (positive altitude is negative Z)
True North [000] is looking down the X-axis
East [090] is looking down the Y-axis
Positive yaw/course is clockwise from North

The NWU system is represented as:

Positive X axis points North
Positive Y axis points West
Positive Z axis points Up (positive altitude is positive Z)
True North [000] is looking down the X-axis
East [090] is looking down the negative Y-axis
Positive yaw/course is counter-clockwise from North

All three systems supported are right-handed.

For all flat earth projections, the local rectilinear coordinate system is a scaled flat earth system centered with the origin at a specified reference latitude, longitude and altitude. Altitude is referenced to height above the WGS-84 ellipsoid. The scaling of the latitude and longitude values into the flat earth system is based on the values of the reference origin. The resulting scaled flat earth only maintains proper scale, direction, distance and area within a short range of the reference origin. The reference origin is specified by either the < **PlatformRefLLA** > or the < **RefLLA** > keywords.

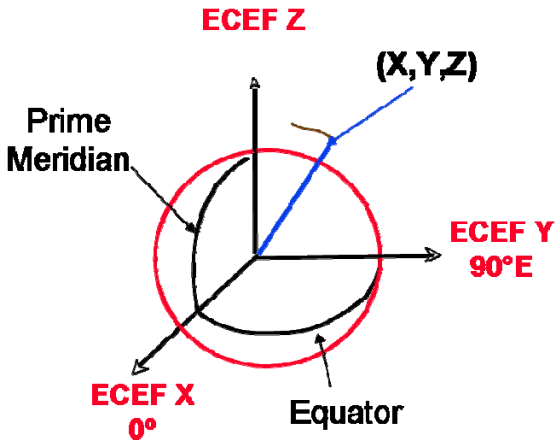
SIMDIS also supports a geodetic coordinate system, LLA. Geodetic coordinates are latitude, longitude, and altitude referenced to the ellipsoidal height (height above the WGS-84 ellipsoid). Latitude ranges are +90.0 (North) to -90.0 (South). Longitude ranges are +180.0 (East) to -180.0 (West). Positive altitude is considered +Z values. If the < **DegreeAngles** > value is set, latitude and longitude can be entered in degrees decimal (D.D), degrees minutes decimal (D M.D) or degrees minutes seconds (DMS), otherwise radians are the input angle unit. If either the degrees minutes decimal or degrees minutes seconds notation is used, the values must be enclosed within quotes, i.e.:

```
// degrees minutes decimal notation
PlatformData    1   0.0 "21 34.88243" "-159 12.47353" 0.0

// degrees minutes seconds notation
PlatformData    1   0.0 "21 34 52.946" "-159 12 28.412" 0.0
```

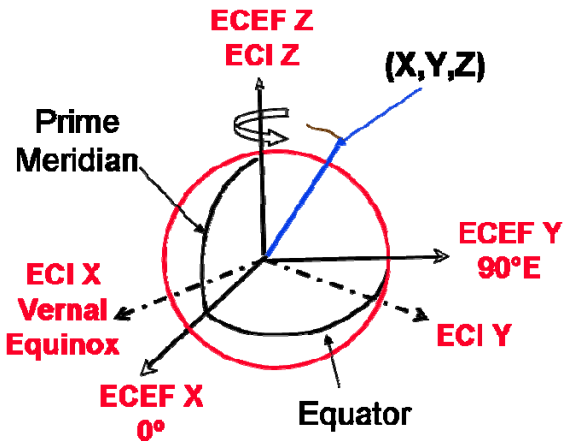
SIMDIS also supports two tangent plane projections, XEAST and Generic Tangent Plane (GTP). Both projections define a local rectilinear coordinate system based on a flat plane tangent to the earth's surface at a specific reference origin. Lines of equal distortion are concentric about the reference origin. Further the distance from the origin, the greater the distortion. The XEAST system is aligned with the Earth such that +X is East, +Y is North, and +Z is up. For the GTP system, the reference origin point does not have to be at this tangential point, but can lie at some other selected point on the x-y plane. Furthermore, the x-y axes do not have to be oriented so that the +y axis points to true north; they can be rotated to any desired angle. Either the keyword < **PlatformTangentPlaneOffset** > or the keyword < **TangentPlaneOffset** > is used to specify the position translation and rotation offset. The "raw" tangent plane values are first translated by x and y, then the rotation is applied. If no offsets are applied, the GTP system becomes a XEAST system. The reference origin is specified by either the < **PlatformRefLLA** > or the < **RefLLA** > keywords.

SIMDIS also supports two geocentric coordinate systems, Earth Centered Earth Fixed (ECEF) and Earth Centered Inertial (ECI). Both the ECEF and ECI systems are referenced to a WGS84 ellipsoidal earth model where (0, 0, 0) is located at the center of mass. For ECEF, the term "Earth Fixed" implies that the axes are fixed with respect to the earth, that is, they rotate with the earth. The Z axis pierces the North Pole, and the XY axis defines the equatorial plane with X pointing to the prime meridian (0 Lon, 0 Lat) and Y pointing east (+90 Lon, 0 Lat).



Earth Center Earth Fixed System

For ECI, The Z axis runs along the Earth's rotational axis pointing north, the X axis points in the direction of the vernal equinox, and the Y axis completes the right-handed orthogonal system. The vernal equinox is an imaginary point in space which lies along the line representing the intersection of the Earth's equatorial (X-Y) plane and the plane of the Earth's orbit around the Sun. The axes defining the ECI coordinate system are "fixed" in space and do not rotate with the Earth.



Earth Centered Inertial System

The keywords < **PlatformReferenceTimeECI** > or < **ReferenceTimeECI** > are used to specify the definition time of the ECI reference frame. If this value is not specified, the time of first data point will be used as the reference time. SIMDIS uses the ECI reference time to calculate an elapsed time which is multiplied by the earth's angular velocity in order to determine the corresponding ECEF position.

All orientations are based on a corresponding right-handed system. Euler angles in SIMDIS follow the aerospace yaw, pitch and roll notation. When either ECEF or ECI Euler angles are converted for use in SIMDIS, the following conversions are used. A locally level, tangential to the earth's surface, north east down (NED) frame is created at the location of the platform. The following direction cosine matrix (DCM) is used to perform the transformation from the inertial ECEF frame to the geographic NED reference frame:

$$T_{I2G} = \begin{bmatrix} -\sin u \cos l & -\sin l & -\cos u \cos l \\ -\sin u \sin l & \cos l & -\cos u \sin l \\ \cos u & 0 & -\sin u \end{bmatrix}$$

Where u = latitude, l = longitude

The platform body frame is aligned X-forward, Y-Starboard (right) and Z-down. The sequence of rotations conventionally used in the aerospace industry to describe the instantaneous attitude with respect to a reference frame (NED) is as follows:

Rotate about the Z-axis, nose right (positive yaw ψ)

Rotate about the new Y-axis, nose up (positive pitch θ)

Rotate about the new X-axis, right wing down (positive roll ϕ)

The following DCM is used to perform the transformation from the geographic (NED) reference frame to the platform body frame:

$$T_{G2B} = \begin{bmatrix} \cos\theta\ \cos\psi & \cos\theta\ \sin\psi & -\sin\theta \\ -\cos\phi\ \sin\psi\ +\sin\phi\ \sin\theta\ \cos\psi & \cos\phi\ \cos\psi\ +\sin\phi\ \sin\theta\ \sin\psi & \sin\phi\ \cos\theta \\ \sin\phi\ \sin\psi\ +\cos\phi\ \sin\theta\ \cos\psi & -\sin\phi\ \cos\psi\ +\cos\phi\ \sin\theta\ \sin\psi & \cos\phi\ \cos\theta \end{bmatrix}$$

Where ψ = yaw, θ = pitch, ϕ = roll

The overall transformation from the inertial (ECEF) frame to the platform body frame is denoted by the following matrix:

$$T_{I2B} = T_{I2G} T_{G2B}$$

Conversely, if moving from the platform-body frame to the inertial frame, the rotations are reversed such that a roll, pitch and yaw sequence is followed. This is accomplished by multiplying the transpose of the above matrices.

A.2.2 Beam Input Data

Two keywords that are [**REQUIRED**] to be specified during the beam data input phase. A third keyword, BeamTargetIDCmd is required if a beam target command is used. The keywords are as follows:

```
BeamOnOffCmd    102  1.000      1
```

Case 1: Azimuth, elevation and range

```
BeamData    102   1.000 yellow 0.1745329 0.1 10000.0
```

Case 2: Pointing values and range determined from tracked target.

NOTE: "target" is a keyword, not the name of a platform. The BeamTargetIDCmd is used to specify the beam's "target".

```
BeamData    103 1.000 yellow "target"

BeamTargetIDCmd   103 5.000 101
```

Case 3: Azimuth and elevation offset values added to body orientation of host platform.

NOTE: "body" is a keyword, not the name of the host platform.

```
BeamData 105 1.000 yellow 0.1745329 0.1 10000.0 "body"
```

Where the < **BeamOnOffCmd** > keyword specifies the following:

Beam unique id to which the data belongs
Time for which this data is valid
Beam state (0: off, 1: on)

Where the < **BeamData** > keyword specifies the following:

Beam unique id to which the data belongs
Time for which this data is valid
Beam color (hexadecimal: 0xAABBGGRR or color string)

After the color field, the remaining fields are based on one of three possible cases. All angle units are radians unless the scenario keyword < **DegreeAngles** > is set to 1.

Case 1:

Azimuth (rad or deg) value measured clockwise from north.
Elevation (rad or deg) value, positive values are measured above the horizon.
Range in meters of beam from host platform.

Case 2:

The "target" keyword is specified as the beam's data. The beam's targeted platform is set using the < **BeamTargetIDCmd** >. The azimuth, elevation and range between the beam's host platform and the targeted platform is then computed.

Case 3:

Azimuth (rad or deg) offset value referenced to host platform's yaw.
Elevation (rad or deg) offset value referenced to host platform's pitch.
Range (meters) of beam from host platform.
The "body" keyword is specified as the beam's data. The inertial yaw and pitch of the host platform are used as the beam's azimuth and elevation.

Where the < **BeamTargetIDCmd** > keyword specifies the following:

Beam unique id to which the data belongs
Time for which this data is valid
Tracked Target's unique id

NOTES:

All angle units are radians unless the scenario keyword < **DegreeAngles** > is set.

BeamTargetCmd is deprecated as of ASI version 8.0, < **BeamTargetIDCmd** > replaces it.

See the Time Formats and Color Formats sections for more details.

A.2.3 Gate Input Data

There are four possible [**REQUIRED**] keywords that can used for gate input data. Three of the keywords are mutually exclusive. The keywords are as follows:

```
GateOnOffCmd  201  1.000  1
```

Case 1: Azimuth, elevation and range

```
GateData      201  1.000  blue 2.1 0.15 0.5 0.5 10000. 11000. 10500.
GateData      201  3.000  blue 2.1 0.15 -1.0 –1.0 10000. 11000. 10500.
```

Case 2: Pointing values and range determined from tracked target.

NOTE: The BeamTargetIDCmd is used to specify the gate's "target".

```
TargetGateData   202  1.000   blue   0.5   0.5   -10.0   50.0    5.0
TargetGateData   202  3.000   blue   -1.0  –1.0  -10.0   50.0    5.0
```

Case 3: Azimuth and elevation offset values added to body orientation of host platform.

```
BodyGateData  203  1.000  blue 0.0 0.0 0.5 0.5 10000. 11000. 10500.
BodyGateData  203  3.000  blue 0.0 0.0 -1.0 –1.0 10000. 11000. 10500.
```

Where the < **GateOnOffCmd** > keyword specifies the following:

Gate unique id to which the gate data belongs
Time for which this gate data is valid
Gate state (0: off, 1: on)

Where the < **GateData** > keyword specifies the following:

Gate unique id to which the data belongs
Time for which this data is valid
Gate color (hexadecimal: 0xAABBGGRR or color string)
Azimuth (rad or deg) value, positive values measured clockwise from north.
Elevation (rad or deg) value, positive values are measured above the horizon.
Angular width (rad or deg) of gate.
Angular height (rad or deg) of gate.
Start position (meters) of gate relative to host platform.
End position (meters) of gate relative to host platform.
Center position (meters) of gate relative to host platform.

Where the < **TargetGateData** > keyword specifies the following:

Gate unique id to which the data belongs
Time for which this data is valid
Gate color (hexadecimal: 0xAABBGGRR or color string)
Angular width (rad or deg) of gate.
Angular height (rad or deg) of gate.
Start position (meters) of gate relative to target platform.
End position (meters) of gate relative to target platform.
Center position (meters) of gate relative to target platform.

Where the < **BodyGateData** > keyword specifies the following:

Gate unique id to which the data belongs
Time for which this data is valid
Gate color (hexadecimal: 0xAABBGGRR or color string)
Azimuth (rad or deg) body offset, measured clockwise from north.
Elevation (rad or deg) body offset, positive values are measured above the horizon.
Angular width (rad or deg) of gate.
Angular height (rad or deg) of gate.
Start position (meters) of gate relative to host platform.
End position (meters) of gate relative to host platform.
Center position (meters) of gate relative to host platform.

NOTES:

All angle units are radians unless the scenario keyword < **DegreeAngles** > is set.

The < **BeamTargetIDCmd** > is used to set to a valid target for both the target beam and target gate data commands. If no target is found, the gate target data is ignored.

If the gate's angular width is specified as a value <=0, the value will be replaced by the host beam's horizontal beam width.

If the gate's angular height is specified as a value <=0, the value will be replaced by the host beam's vertical beam width.

The center value for the target gate data command is optional. If it is not specified, SIMDIS will generate a center value between the start and end positions.

See the Time Formats and Color Formats sections for more details.

A.2.4 Generic Input Data

There is only one keyword that is [**REQUIRED**] to be specified during the generic data input phase. However, generic data is to be input as pairs of tags and data. The tag is used to describe the type of data that will come next. An example data section is as follows:

```
GenericData    601   "SHIPEVENT"      "ship moves forward"  1.0   5.0
GenericData    601   "SHIPEVENT"      "ship moves backward" 10.0  2.0
GenericData    602   "MISSILEEVENT"   "bogey : turns on"    0.0
GenericData    602   "MISSILEEVENT"   "bogey : looks for target"  2.0
GenericData    602   "MISSILEEVENT"   "bogey : 2nd gate appears"  4.0
```

Where the < **GenericData** > keyword specifies the start of generic related data:

Unique ID of a platform, beam, or gate (or 0)	[**REQUIRED**]
Data field tag, (char string)	[**REQUIRED**]
Data field description, (char string)	[**REQUIRED**]
Begin time for which this data is valid	[**REQUIRED**]
Expiration Time (sec) offset (+) from begin time (double)	[**OPTIONAL**]

The Unique ID specified must correspond to the unique ID of a platform, beam, or gate entity. A special value of 0 can be used to specify that the generic data element is associated with the scenario itself. In addition, a < **GenericData** > command may appear before the corresponding < **PlatformID** >, < **BeamID** >, or < **GateID** > command in an ASI file.

NOTE: In ASI version 8, the initialization of Generic Data Objects has been removed. The commands GenericObjId and GenericObjectRefId are deprecated as of ASI file format version 7.

In order for the begin time to be valid, it must fall with in the beginning and ending bounds of the reference object.

Expiration time is a positive offset from the begin time instead of an absolute time. If an expiration time is not specified, it defaults to an infinite time, i.e. –1. The following examples have the same expiration time:

```
GenericData  602 "MISSILEEVENT" "bogey: second gate appears" 4.0
GenericData  602 "MISSILEEVENT" "bogey: second gate appears" 4.0 -1
```

Only the expiration time may have a value of –1. Specifying a –1 for the start time is invalid and will generate an error.

To remove or clear a specific generic data tag from being displayed, use the " ClearGenData " keyword in the data field. The following example removes any "MISSILEEVENT" generic data prior to 6.0. Furthermore, expiration time is not used when a " ClearGenData " data tag is processed.

```
GenericData  602   "MISSILEEVENT" "bogey : second gate appears"  4.0
GenericData  602   "MISSILEEVENT" "ClearGenData"    6.0
```

NOTE: Unlike data with finite expiration times, there can only be one unique data field tag per begin time for generic data with an infinite expiration (-1). For example, the following will not work:

```
GenericData   300    "SIMDIS_TitleText"
              "Detect`50`12`0`1`0xff0000ff`arialbd.ttf 20" 30.0 -1
GenericData   300    "SIMDIS_TitleText"
              "Launch`50`24`0`1`0xff0000ff`arialbd.ttf 20" 30.0 -1
```

Only the first generic data value encountered will be used, all subsequent generic data with the same begin time will be ignored. Furthermore, infinite generic data with the same data field tag and an older begin time will overwrite any previous data.

Category Input Data

The following are examples of category data lines:

```
CategoryData    601    0     "Platform Type"    "Aircraft"
CategoryData    601    4     "Platform Type"    "Helicopter"
CategoryData    601    0     "Affinity"         "Unknown"
CategoryData    601    4     "Affinity"         "Friendly"
CategoryData    601    -1    "Data Source"      "LATR"
```

Where the <**CategoryData**> keyword specifies the start of category related data:

Unique ID of a platform, beam, or gate	[**REQUIRED**]
Time for which this data is valid	[**REQUIRED**]
Category Name, (char string)	[**REQUIRED**]
Category Value, (char string)	[**REQUIRED**]

The Unique ID specified must correspond to the unique ID of a platform, beam, or gate entity. The category value may not be “Unlisted Value” or “No Value”. Also, the use of special characters in the category name or value is not allowed. The following characters may not be used in the category name or value: #, ‘,`,~.

A.3 Miscellaneous Formats and Data

The following sections are the "catchall" section for the ASI File Format. Here you will find information on various formats that the ASI importer supports.

A.3.1 Time Formats

SIMDIS assumes all times are referenced to the beginning of a calendar year. Time can be specified in the following formats:

SS.SSS
"MM:SS.SSS"
"HH:MM:SS.SSS"
"DDD YYYY HH:MM :SS.SSS"
"Month Day YYYY HH:MM :SS.SSS"

If time is not represented as a single value, then encompassing quotes are required. Furthermore, if < **ReferenceYear** > is not set, the current year is determined from the host computer's system clock.

The time formats have changed as of ASI file format version 9. Ordinal-formatted time values (DDD, day of the year 001-366) are now saved with a four-digit year. In Month-formatted time values, the four-digit year is now located after "Month Day", rather than at the end of the string. Old ASI time formats are still supported, but will not be saved out by any SIMDIS application that generates ASI format files. This change was made to allow for more precise definition of time values outside the current reference year. However, time values must still be positive, and must still occur after midnight January 1 of the reference year.

NOTE: For static entities, only a single data value is needed. Replace the time value with a -1.0. For example:

PlatformData 1 **- 1.0** 1115.0 342.0 0.0 3.344 0.0 0.0 10.0

Otherwise SIMDIS requires all platforms to have specific begin and end times that are NOT equal. i.e., there must be at least two < **PlatformData** > values in order for the platform to be valid.

A.3.2 Color Formats

All colors are specified as either 0xAABBGGRR, where

Value	Meaning
AA	Alpha component, range 00 to FF (0-255) in hexadecimal
BB	Blue component, range 00 to FF (0-255) in hexadecimal
GG	Green component, range 00 to FF (0-255) in hexadecimal
RR	Red component, range 00 to FF (0-255) in hexadecimal

Or the following character strings can be used. The hexadecimal numbers are the exact color each string is mapped to.

Value	Meaning
blue	0x80ff0000
red	0x800000ff
green	0x8000ff00
white	0x80ffffff
yellow	0x8000ffff
purple	0x80ff00ff
magenta	0x808500ff
cyan	0x80ffff00
black	0x80000000
brown	0x80105050
orange	0x800080ff
gray	0x80808080

A.3.3 ASCII Antenna Pattern File Formats

Table Antenna Pattern (.aptf)

This antenna file contains a series of sub-tables of antenna patterns based on their respective symmetry. The angular data can be to any desired degree of resolution and can be irregularly spaced. The data within each sub-table should be in increasing angular order, e.g., 0 to 180 degrees. For this particular antenna pattern, the following four sub-tables define the pattern:

```
1: [-180 ,0 ] azimuth (H-plane)
2: [0,180]    azimuth (H-plane)

3: [-90 ,0 ]  elevation (E-plane)
4: [0 ,90 ]   elevation (E-plane)
```

The antenna symmetry value indicates the number of tables the user is going to provide. If the symmetry is 1, then the user will provide the [0, 180] azimuth table. The program will reuse this table for the other three tables. If the symmetry is 2, then the user will provide the [0, 180] azimuth table and the [0, 90] elevation table. The program will reuse these tables for the missing azimuth, and elevation tables. If the symmetry is 4, the user will provide all four azimuth and elevation tables.

For symmetric antenna patterns, Symmetry = 2, tables should cover the sectors [0,180] for azimuth patterns and [0 ,90] for elevation patterns. Internally, mirroring the data about the zero axes creates a 360-degree azimuth and a 180-degree elevation pattern. Sub-tables should have the following order:

```
1: [0,180]   azimuth (H-plane)
2: [0 ,90 ]  elevation (E-plane)
```

For asymmetric antenna patterns, Symmetry = 4, all four sub-tables are required. These four tables must be input sequentially and cover the sectors in the following order:

```
1: [-180 ,0 ] azimuth (H-plane)
2: [0,180]    azimuth (H-plane)
3: [-90 ,0 ]  elevation (E-plane)
4: [0 ,90 ]   elevation (E-plane)
```

The first field in the antenna pattern file, "Angle units", represents the units for all angular data. Available formats:

```
0: degrees
1: radians
```

The pattern data is comprised of two columns. The first contains all angular data. The second column contains the antenna pattern gain in dB.

Below is a sample antenna pattern file for a symmetric pattern, i.e. Symmetry = 2. The asymmetric is similar, except the first azimuth pattern range is [-180 ,0], the second table

is the azimuth pattern from [0,180]. Elevation tables are 3 and 4 respectively with ranges [-90 ,0] and [0,90].

The file is delimited on white space (tabs and spaces) and can be fond on the next page.

```
//Sample Data For a Symmetric Pattern of < AntennaPattern > table
0 2               (Angle units (0:deg , 1:rad), Symmetry)
181               (Azimuth Pattern Table Size)
0.000 - 62.400    (angle (deg), gain (dB))
1.000 - 63.200
2.000 - 64.600
3.000 - 63.000
.
.
.
177.000     - 22.000
178.000     - 10.000
179.000     - 3.200
180.000     0.000
91                      (Elevation Pattern Table Size)
0.000 - 54.000    (angle (deg), gain (dB))
1.000 - 52.000
2.000 - 48.800
3.000 - 50.500
.
.
.
87.000      - 33.200
88.000      - 7.200
89.000      - 1.200
90.000      0.000
```

Within SIMDIS linear interpolation is used to determine the gain value based on an input azimuth and elevation value. The input values are then normalized with the vertical and horizontal beam widths in order to return a weighted average antenna gain value.

Relative Table Antenna Pattern (.aprf)

This antenna file contains a series of 2-D tables of antenna patterns based on azimuth (horizontal plane) and elevation (vertical plane) data. The angular data can be to any desired degree of resolution and can be irregularly spaced. It is required that the angles be input in the order of -180 to 180 for azimuth and -90 to 90 for elevation, with the main beam value located at 0.0 degrees. Angular units are in degrees. The corresponding gain data is referenced to the main beam gain. In other words, the maximum gain value should be 0.0, all remaining gain values will be relative to the main gain. Gain units are in dB.

The file is delimited on white space (tabs and spaces).

```
//   Sample Data For < AntennaPattern > anttable
181 91   // # of azimuth data points, # of elevation data points
// azimuth data:  angle (deg), gain (dB)
-180.0            -30.0000000001
-178.0            -30.3103568862
-176.0            -30.4808976808
.
.
.
-4.0        -8.8405020452
-2.0        -2.37823713123
0.0         0.0              // main beam value
2.0         -2.37823713123
4.0         -8.8405020452
.
.
.
176.0       -30.4808976808
178.0       -30.3103568862
180.0       -30.0000000001

// elevation data:  angle (deg), gain (dB)
-90.0       -30.0000000001
-88.0       -30.3103568862
-86.0       -30.4808976808
.
.
.
-4.0        -8.8405020452
-2.0        -2.37823713123
0.0         0.0         // main beam value
2.0         -2.37823713123
4.0         -8.8405020452
.
.
.
86.0        -30.4808976808
88.0        -30.3103568862
90.0        -30.0000000001
```

Within SIMDIS linear interpolation is used to determine the gain value based on an input azimuth and elevation value. The input values are then normalized with the vertical and horizontal beam widths in order to return a weighted average antenna gain value.

Monopulse Antenna Pattern (.apmf)

This antenna file contains a series of monopulse antenna patterns based on frequency, azimuth and elevation data. The patterns are subdivided into two channels, the sum channel and the diff (difference or delta) channel. Both channels have the same format.

The first parameter should be the keyword <sum> or <diff>. This is followed by the frequency information. Frequencies are given in Hz and are in the following format:

start end step

Next is the azimuth data. All angular units are in degrees. It has the same format:

start end step **Note**: -180 deg < Angle(x) < 180 deg

Next is the elevation data. All angular units are in degrees. It has the same format:

start end step **Note**: -90 deg < Angle(x) < 90 deg

All data is required to be evenly spaced.

Finally the remaining data consists of magnitude (dB) and phase (deg) pairs. Below is pseudo code that represents how the data is organized.

The file is delimited on white space (tabs and spaces).

```
// Sample data for < AntennaPattern > monopulse
sum // sum channel keyword (4 frequencies)
8800000000.0 9400000000.0 200000000.0 //freq bgn end step (Hz)
-36.0 36.0  0.5   // azim info:  bgn end step (deg) (144)
-36.0 36.0  0.5   // elev info:  bgn end step (deg) (144)
-27.520000  85.700000   // magnitude (dB) & phase (deg) pairs
-28.210000  86.400000
-29.280000  87.000000
.
.
.
diff  // difference channel keyword  (4 frequencies)
8800000000.0 9400000000.0 200000000.0 // freq bgn end step (Hz)
-36.0 36.0  0.5         // azim info:  bgn end step (deg) (144)
-36.0 36.0  0.5         // elev info:  bgn end step (deg) (144)
-41.670000  125.700000  // magnitude (dB) & phase (deg) pairs
-40.290000 120.200000
-39.130000 127.600000
.
.
.

// pseudo code for organization of magnitude and phase pairs
for (number of frequencies)  // above case would be 4
{
      for (number of azimuths) // above case would be 144
      {
            for (number of elevations) // above case would be 144
            {
                  magnitude(dB)  phase(deg)
```

```
                }
        }
}
```

In the above case, there are (4 x 144 x 144) 82944 magnitude and phase pairs for each channel.

Within SIMDIS bilinear interpolation is used to determine the gain value based on an input azimuth and elevation value. Unlike the table antenna pattern file format, the input values are not normalized with the vertical and horizontal beam widths and the resulting gain is not weighted.

Bilinear Antenna Pattern (.apbf)

This antenna file contains a series of bilinear antenna patterns based on frequency, azimuth and elevation data.

The first parameter should be the keyword <bilinear>. This is followed by the frequency information. Frequencies are given in Hz and are in the following format:

start end step

Next is the azimuth data. All angular units are in degrees. It has the same format:

start end step **Note**: -180 deg < Angle(x) < 180 deg

Next is the elevation data. All angular units are in degrees. It has the same format:

start end step **Note**: -90 deg < Angle(x) < 90 deg

All data is required to be evenly spaced.

Finally the remaining data consists of the gain (dB) values. Below is pseudo code that represents how the data is organized.

The file is delimited on white space (tabs and spaces).

```
// Sample data for < AntennaPattern > bilinear
bilinear // (4 frequencies)
8800000000.0 9400000000.0 200000000.0 //freq bgn end step (Hz)
-36.0 36.0  0.5   // azim info:  bgn end step (deg) (144)
-36.0 36.0  0.5   // elev info:  bgn end step (deg) (144)
-27.520000        // magnitude (dB)
-28.210000
-29.280000
.
.
.

// pseudo code for organization of magnitude data
for (number of frequencies) // above case would be 4
{
      for (number of azimuths) // above case would be 144
      {
```

```
                    for (number of elevations) // above case would be 144
                    {
                          Gain (dB)
                    }
              }
        }
```

In the above case, there are (4 x 144 x 144) 82944 gain values.

Within SIMDIS bilinear interpolation is used to determine the gain value based on an input azimuth and elevation value. Unlike the table antenna pattern file format, the input values are not normalized with the vertical and horizontal beam widths and the resulting gain is not weighted.

National Spectrum Manager's Association (NSMA) Antenna Pattern (.nsm)

This antenna pattern file contains an interpretation of the NSMA standard format for the electronic transfer of antenna patterns (see REP.WG16.89.003) by OETINFO an Engineer at the Federal Communications Commission.

File Naming Convention:

The filename should be a case insensitive, unique, six digit ID code (see WG16.95.043) assigned by the manufacturer with the file extension of ".nsm". (e.g. 24032g.nsm) 1st digit: [0-9, 1 digit Frequency Code Number] + 2nd-5th digits: [0000-ZZZ Z, 4 digit alpha-numeric unique Manufacturer ID Number assigned by manufacturer] +6th digit: [A|C|G|M; 1 digit Manufacturer Code which is registered with the NSMA] + Extension: [.nsm, a standard file extension for this version (v1) of the NSMA standard]

File Structure:

```
[Antenna Manufacturer] + CRLF
[Antenna Model number] + CRLF
[Comment] + CRLF
[FCC ID number] + CRLF
[reverse pattern ID number] + CRLF
[date of data] + CRLF
[Manufacturer ID Number (see file naming convention)] + CRLF
[frequency range] + CRLF
[mid-band gain] + CRLF
[Half-power beam width] + CRLF
[polarization (char 7) + chr$(32) + data count (char 7)  + chr$(32) +
CRLF]
[angle(1) (char 7) + chr$(32) + relative gain in dB(char 7) + chr$(32)
+ CRLF]
.
.
.
[angle(data count) (char 7) + chr$(32) + relative gain in dB (char 7) +
chr$(32) + CRLF]
.
.
.
```

```
[polarization (char 7) + chr$(32) + data count (char 7)  + chr$(32) +
CRLF]
[angle(1) (char 7) + chr$(32) + relative gain in dB(char 7) + chr$(32)
+ CRLF]
.
.
.
[angle(data count) (char 7) + chr$(32) + relative gain in dB (char 7) +
chr$(32) + CRLF]
```

The first ten lines must exist and end with both a carriage return and line feed.

Dates should be formatted (12/31/[19]96 or 31 DEC [19]96 or [19]96.12.31).

Mid-band Gain should be in dBi (relative to and isotropic radiator).

Frequency Range must be in Megahertz. (6525.0 - 6875.0)

Place "NONE" on any rows which otherwise would be blank.

All values should be left justified in their character fields.

Polarization must be in the set [HH|HV|VV|VH|ELHH|ELHV|ELVV|ELVH]

-180 deg < Angle(x) < 180 deg for [HH|HV|VV|VH]

-90 deg < Angle(x) < 90 deg for [ELHH|ELHV|ELVV|ELVH]

Angle (1) < Angle (2) < ... < Angle (data count)

Relative Gain in dB < ~0 including sign
(Most values will be non positive. Exceptions may include antennas which have slightly depressed main lobes at 0 degrees. E.g. some panel antennas) (relative to the mid-band gain given on line 9 above)

Details from NSMA WG-16 publication

The following are detailed explanations of each of the data lines.

`(Antenna Manufacturer)` `(30 data)`

This is the name under which the data was filed with the FCC. There will be no abbreviations.

`(Full model number)` `(30 data)`

This is the full model number as used when the data was filed with the FCC. Modifiers to the model number such as dashes or exceptions are to be included.

`(FCC ID number)` `(16 data)`

This is the ID number issued by the Common Carrier Branch of the FCC. For services which do not issue ID numbers, insert the word (none) in upper case.

`(Reverse pattern ID number)` `(16 data)`

This lists the reverse pattern FCC ID number. The reverse pattern is generally obtained by inserting the feed in a opposite manner in order to reverse the pattern.

`(date of data)` `(16 data)`

This date is the date referenced on the published pattern

`(manufacturer ID number)` `(4 data)`

This is the reference number assigned by the antenna manufacturer. This 4 digit alpha-numeric should be included with all antennas models and should also be used to name the antenna pattern filename.

`(frequency range)` `(16 data)`

This is to identify the full frequency range for which this pattern is valid and agrees with the range as specified in the printed pattern. The frequency is in Megahertz.

`(mid-band gain)` `(16 data)`

This is the gain of the antenna at mid-band. The gain is in gain above an isotropic radiator (dBi).

`(half power beam width)` `(16 data)`

This is the included angle centered on the main beam of the antenna and defines the angle where the antenna response falls -3 dB.

```
(polarization) (data count)   (7 data, 1 space)      (7 data, 1 space)
```

The data is preceded by an indication of the polarization the data. The commonly accepted polarization designators for linear polarization are to be used:

HH
: Horizontal polarized port response to a horizontally polarized signal in the horizontal direction.

HV
: Horizontal polarized port response to a vertically polarized signal in the horizontal direction.

VV
: Vertical polarized port response to a vertically polarized signal in the horizontal direction

VH
: Vertical polarized port response to a horizontally polarized signal in the horizontal direction

ELHH
: Horizontal polarized port response to a horizontally polarized signal in the vertical direction

ELHV
: Horizontal polarized port response to a vertically polarized signal in the vertical direction

ELVV
: Vertical polarized port response to a vertically polarized signal in the vertical direction

ELVH
: Vertical polarized port response to a horizontally polarized signal in the vertical direction

The data count will be the number of data points to follow.

All eight responses should be included. If different polarizations have identical responses, they are to be duplicated in order that a full set of data be listed.

```
(angle)        (response)     (7 data, 1 space)      (7 data, 1 space)
```

A full compliment of data will show the antenna response in the horizontal direction for a 'horizontal cut' and in the vertical direction for a 'vertical cut'.

The data is presented in two columns. The angle of observation is listed first followed by the antenna response.

For the horizontal direction, the angle of observation starts from -180 degrees (defined as the left side of the antenna) and decrease in angle to the main beam, 0 degrees, and then increase to +180 degrees. The full data will cover the 360 degrees of the antenna.

For the vertical direction, the angle of observation starts from -5 (-90) degrees (defined as the antenna response below the main beam) and decrease in angle to the main beam, 0 degrees, and then increase to +5 (+90) degrees. The full data will cover the 10 (180) degrees centered about the main beam.

The antenna response is listed as dB down from the main lobe response and is shown as negative.

As a minimum, the data points are the breakpoints. That is, those points which define a change in the slope of the data or an adequate number of points to define a non-linear line. It is acceptable to include periodical points (e.g. every 1 degree or more) between the breakpoints.

Within SIMDIS linear interpolation is used to determine the gain value based on an input azimuth and elevation value. Unlike the table antenna pattern file format, the input values are not normalized with the vertical and horizontal beam widths and the resulting gain is not weighted. Furthermore, the mid band gain and half power beam widths found in the NSMA file are used in place of the values found in the ASI file for determining the gain response of the antenna. In other words, modifying the main lobe gain value in the Super Form Beam Tab will have no effect on the gain value returned from this type of antenna pattern.

A.3.4 ASCII Radar Cross-Section File Formats

Lookup Table (LUT) Radar Cross-Section Format

This radar cross-section (RCS) file has the ability to contain multiple sub-tables of radar cross sections, with each RCS based on a measured frequency and elevation angle. The azimuthal data can be to any desired degree of resolution and can be irregularly spaced. This table also provides the ability to perform various types of distribution functions on the RCS. Currently, Mean, Gaussian, Rayleigh and Log Normal distributions are supported. This table also supports a symmetric RCS pattern about the azimuth axis if 2 is selected as the table type. Below is a sample RCS lookup table:

```
0               RCS Lookup Table File Designation
RCS Test Case Data (RCS File Description)
1               Table Type (0=distribution, 1=LUT, 2=symmetric)
0               Func (0=Mean, 1=Gaussian, 2=Rayleigh, 3=Log Normal)
0.0             Scintillation Modulation (dBsm)
2               Number of Tables
8000.0          Frequency Table 1 (Mhz)
   0.0          Elev (deg)
   0            Polarity (0=Horizontal, 1=Vertical, 2=Circular)
 360            Number of Aspect Data Elements
0         0     Angle Units (0=deg, 1=rad) RCS Units (0=sq m, 1=dBsm)
0.02      8.02
1.00      7.88
...
358.00    4.76
358.98    8.35
8000.0          Frequency Table 2 (Mhz)
   10.0         Elev (deg)
   0            Polarity (0=Horizontal, 1=Vertical, 2=Circular, etc)
 360            Number of Aspect Data Elements
0         0     Angle Units (0=deg, 1=rad) RCS Units (0=sq m, 1=dBsm)
0.02      8.02
1.00      7.88
...
358.00    4.76
358.98    8.35
```

Ship Air Defence Model (SADM) Radar Cross-Section Format

This SADM radar cross-section (RCS) file has the ability to contain multiple sub-tables of radar cross sections, with each RCS based on a measured frequency and elevation angle. The azimuthal data can be to any desired degree of resolution and can be irregularly spaced. SADM RCS files can have up to 361 azimuth values, and up to 16 elevation values, up to 16 frequencies, and must have both horizontal and vertical polarizations. The frequency values must be monotonically increasing, but the polarization values can occur in either order. Following is a sample SADM RCS table.

```
%*********************************************************
% This is a sample RCS file.  Comments prefaced by a "%"
% symbol can be included before the "RCS" keyword to
% document the contents of this file.  To be safe, DO
% NOT include the keyword "&+RCS" anywhere within these
% comments.
```

```
%
% The initial parameters that go in this file are:
%
% RCS_FREQ_INTERPOLATE = T/F
% RCS_N_AZ =
% RCS_N_EL =
% RCS_N_FREQ =
% RCS_EL =   el1   el2  ...
%
% They are followed by the following values, repeated for
% each frequency / polarization combination.
%
% RCS_FREQ = freq1
% RCS_POL = H
% RCS_ON_AXIS = on_axis_rcs @ freq1 & H pol
% RCS_TABLE =
%   ...
%   ...
% RCS_FREQ = freq1
% RCS_POL = V
% RCS_ON_AXIS = on_axis_rcs @ freq1 & V pol
% RCS_TABLE =
%   ...
%   ...
% RCS_FREQ = freq2
% RCS_POL = H
% RCS_ON_AXIS = on_axis_rcs @ freq2 & H pol
% RCS_TABLE =
%   ...
% RCS_FREQ = freq2
% RCS_POL = V
% RCS_ON_AXIS = on_axis_rcs @ freq2 & V pol
% RCS_TABLE =
%   ...
%   ...
% etc
% /            (terminate with a slash)
%
&RCS
RCS_FREQ_INTERPOLATE = T
RCS_N_AZ = 73
RCS_N_EL = 3
RCS_N_FREQ = 3
RCS_EL = 0 30 60
RCS_FREQ = 9
RCS_POL = V
RCS_ON_AXIS = -100
RCS_TABLE =
  0.0000 44.0000 42.1000 40.1000
  5.0000 38.0000 35.8000 33.8000
 10.0000 40.0000 37.8000 35.8000
.
.
.
355.0000 35.0000 32.8000 30.8000
360.0000 41.0000 39.1000 37.1000
```

A.3.5 SIMDIS Scenario Specific Generic Data Formats

Instead of relying on a purely textual representation, each of the following scenario level generic data commands performs a unique graphics based function within SIMDIS. All reserved generic data keywords begin with the text "SIMDIS_". Additionally, plug-ins can store text data in a scenario by adding generic data that begins with the text "PluginData_".

Title Text

```
GenericData      0      "SIMDIS_TitleText"      "COMEX`50`20`0`1"
"00:01:05.000" "10"
GenericData      0      "SIMDIS_TitleText"      "FINEX`50`20`0`1
`0xFF808080`arial.ttf 40" "00:09:04.000"
```

The format for the tag and data are as follows:

Where the keyword < SIMDIS_TitleText > is the specific generic data tag. Notice the data fields are delimited by a back tick < ` >. SIMDIS_TitleText data contains 7 fields, of which only 5 are required:

Display label, (char string)	[**REQUIRED**]
Percentage of screen X position, (integer)	[**REQUIRED**]
Percentage of screen Y position, (integer)	[**REQUIRED**]
Horizontal screen justification, (integer)	[**REQUIRED**]
Vertical screen justification, (integer)	[**REQUIRED**]
Label color (unsigned integer) (0xAABBGGRR)	[**OPTIONAL**]
Label font (char string)	[**OPTIONAL**]

The display label is automatically center justified. The color (0xAABBGGRR) defaults to 0xffffffff. The font defaults to arialbd.ttf 50. Available options for horizontal screen justification are 0 for left and 1 for right. Available options for vertical screen justification are 0 for bottom and 1 for top.

Screen Text

```
GenericData  0  "SIMDIS_ScreenText"  "Red  Color  Change`50`85`1
`0xff0000ff`arialbd.ttf 30" "10:04:10.000" "10"
GenericData  0  "SIMDIS_ScreenText"  "Orig  Color  Change`50`85`1"
"10:07:10.000" "10"
```

The format for the tag and data are as follows:

Where the keyword < SIMDIS_TitleText > is the specific generic data tag. Notice the data fields are delimited by a back tick < ` >. SIMDIS_TitleText data contains 6 fields, of which only 4 are required:

Display label (char string)	[**REQUIRED**]
Percentage of screen X position (integer)	[**REQUIRED**]
Percentage of screen Y position (integer)	[**REQUIRED**]
String justification (integer)	[**REQUIRED**]
Label color (unsigned integer) (0xAABBGGRR)	[**OPTIONAL**]
Label font (char string)	[**OPTIONAL**]

The horizontal screen justification is left. The vertical screen justification is bottom. The color (0xAABBGGRR) defaults to 0xffffffff. The font defaults to arialbd.ttf 20. Available options for string justification are 0 for left, 1for center, and 2 for right

Info Text

```
GenericData    0    "SIMDIS_InfoText"    "3rd    Stage    Deployed"
"10:03:10.000" "10"
GenericData      0      "SIMDIS_InfoText"      "KW      Deployed
`0xff00ff00`arialbd.ttf 20" "10:06:10.000" "10"
```

The format for the tag and data are as follows:

Where the keyword < SIMDIS_InfoText > is the specific generic data tag. Notice the data fields are delimited by a back tick < ` >. SIMDIS_InfoText data contains three fields, of which only one is required:

Display label, (char string)	[**REQUIRED**]
Label color (unsigned integer) (0xAABBGGRR)	[**OPTIONAL**]
Label font (char string)	[**OPTIONAL**]

The display label is automatically center justified. The screen placement of the label is fixed to 75 pixels up from the bottom, in the center of the screen. The horizontal screen justification is left. The vertical screen justification is bottom. The color (0xAABBGGRR) defaults to 0xffffffff. The font defaults to arialbd.ttf 35.

Examples in the use of the various generic data text commands can be found in the following distributed ASI files:

$(SIMDIS_DIR)/demos/SIMDIS/import/sampleGenericData2.asi
$(SIMDIS_DIR)/demos/SIMDIS/Users/NUWC/TorpInternal_good.asi

A.3.6 SIMDIS Platform Specific Generic Data Formats

Instead of relying on a purely textual representation, each of the following platform level generic data commands performs a unique graphics based function within SIMDIS.

Call Sign

```
GenericData 602 "SIMDIS_Callsign" "Missile" "10:01:05.000"
GenericData 602 "SIMDIS_Callsign" "Missile" "10:03:09.900"
GenericData 602 "SIMDIS_Callsign" "TSRM" "10:03:10.000"
GenericData 602 "SIMDIS_Callsign" "TSRM" "10:06:09.900"
GenericData 602 "SIMDIS_Callsign" "KW" "10:06:10.000"
```

The format for the tag and data are as follows:

Where the keyword < SIMDIS_Callsign > is the specific generic data tag. SIMDIS_Callsign data contains only 1 required data field:

Call sign (char string) [**REQUIRED**]

This generic data tag is used to change the label for the referenced platform, modifying the same value that PlatformName modifies. Notice how the data is repeated in a time bracket. This must be done in order to preserve the correct call sign if the data is to be played backwards or is to loop back on it self.

Examples in the use of the SIMDIS_Callsign generic data command can be found in the following distributed ASI files:

$(SIMDIS_DIR)/demos/SIMDIS/import/sampleGenericData2.asi
$(SIMDIS_DIR)/demos/SIMDIS/Users/NUWC/TorpInternal_good.asi

Change Icon

```
GenericData 602 "SIMDIS_ChangeIcon" "sm-3" "10:01:05.000"
GenericData 602 "SIMDIS_ChangeIcon" "sm-3" "10:03:09.900"
GenericData 602 "SIMDIS_ChangeIcon" "sm-3_3rd_stage" "10:03:10.000"
GenericData 602 "SIMDIS_ChangeIcon" "sm-3_3rd_stage" "10:06:09.900"
GenericData 602 "SIMDIS_ChangeIcon" "sm-3_kw" "10:06:10.000"
```

The format for the tag and data are as follows:

Where the keyword < SIMDIS_ChangeIcon > is the specific generic data tag. SIMDIS_ChangeIcon data contains only 1 required data field:

3-D model filename, (char string) [**REQUIRED**]

This generic data tag is used to change the 3-D icon for the referenced platform. Notice how the data is repeated in a time bracket. This must be done in order to preserve the correct 3-D icon if the data is to be played backwards or is to loop back on it self.

Examples in the use of the SIMDIS_ChangeIcon generic data command can be found in the following distributed ASI files:

$(SIMDIS_DIR)/demos/SIMDIS/import/sampleGenericData2.asi
$(SIMDIS_DIR)/demos/SIMDIS/Users/SADM/simdis_pres_20008_0001.asi

Dynamic Scale

```
GenericData 602 "SIMDIS_DynamicScale" "0" "10:01:05.000"
GenericData 602 "SIMDIS_DynamicScale" "0" "10:03:09.900"
GenericData 602 "SIMDIS_DynamicScale" "1" "10:03:10.000"
```

The format for the tag and data are as follows:

Where the keyword < SIMDIS_DynamicScale > is the specific generic data tag. SIMDIS_DynamicScale data contains only 1 required data field:

On/Off flag (char string) 1=on, 0=off [**REQUIRED**]

This generic data tag is used to toggle the dynamic scaling for the referenced platform. Notice how the data is repeated in a time bracket. This must be done in order to preserve the correct dynamic scaling if the data is to be played backwards or is to loop back on it self.

Examples in the use of the SIMDIS_DynamicScale generic data command can be found in the following distributed ASI files:

$(SIMDIS_DIR)/demos/SIMDIS/Users/SADM/simdis_pres_60001_0001.asi
$(SIMDIS_DIR)/demos/SIMDIS/Users/SADM/simdis_pres_20008_0001.asi

Font Color

```
# Sets the font color to it's original value
GenericData 602 "SIMDIS_FontColor" "-1" "10:01:05.000"
GenericData 602 "SIMDIS_FontColor" "-1" "10:04:09.900"
# Set the font color to green (0xAABBGGRR) in hex
GenericData 602 "SIMDIS_FontColor" "0xFF00FF00" "10:04:10.000"
GenericData 602 "SIMDIS_FontColor" "0xFF00FF00" "10:07:09.900"
# Sets the font color to it's original value
GenericData 602 "SIMDIS_FontColor" "-1" "10:07:10.000"
```

The format for the tag and data are as follows:

Where the keyword < SIMDIS_FontColor > is the specific generic data tag. SIMDIS_FontColor data contains only 1 required data field:

Font color (char string) [**REQUIRED**]

This generic data tag is used to change the label's font color for the referenced platform. Notice how the data is repeated in a time bracket. This must be done in order to preserve the correct font color if the data is to be played backwards or is to loop back on it self.

Examples in the use of the SIMDIS_FontColor generic data command can be found in the following distributed ASI file:

$(SIMDIS_DIR)/demos/SIMDIS/import/sampleGenericData2.asi

Override Color

```
# Sets the platform's color to it's original value
GenericData 602 "SIMDIS_OverrideColor" "-1" "10:01:05.000"
GenericData 602 "SIMDIS_OverrideColor" "-1" "10:04:09.900"
# Set the platform color to red (0xAABBGGRR) in hex
GenericData 602 "SIMDIS_OverrideColor" "0xFF0000FF" "10:04:10.000"
GenericData 602 "SIMDIS_OverrideColor" "0xFF0000FF" "10:07:09.900"
# Sets the platform's color to it's original value
GenericData 602 "SIMDIS_OverrideColor" "-1" "10:07:10.000"
```

The format for the tag and data are as follows:

Where the keyword < SIMDIS_OverrideColor > is the specific generic data tag. SIMDIS_OverrideColor data contains only 1 required data field:

Override color (char string) [**REQUIRED**]

This generic data tag is used to change the override color for the referenced platform, but it only applies to 3-D models. Notice how the data is repeated in a time bracket. This must be done in order to preserve the correct override color if the data is to be played backwards or is to loop back on it self.

Examples in the use of the SIMDIS_OverrideColor generic data command can be found in the following distributed ASI files:

$(SIMDIS_DIR)/demos/SIMDIS/import/sampleGenericData2.asi
$(SIMDIS_DIR)/demos/SIMDIS/import/errorEllipse.asi

Scale Level

```
GenericData 602 "SIMDIS_ScaleLevel" "1" "10:01:05.000"
GenericData 602 "SIMDIS_ScaleLevel" "1" "10:03:09.900"
GenericData 602 "SIMDIS_ScaleLevel" "3" "10:03:10.000"
```

```
GenericData 602 "SIMDIS_ScaleLevel" "3" "10:06:09.900"
GenericData 602 "SIMDIS_ScaleLevel" "5" "10:06:10.000"
```

The format for the tag and data are as follows:

Where the keyword < SIMDIS_ScaleLevel > is the specific generic data tag. SIMDIS_ScaleLevel data contains only 1 required data field:

Scale Level (char string) [**REQUIRED**]

This generic data tag is used to change the scale level for the referenced platform. Notice how the data is repeated in a time bracket. This must be done in order to preserve the correct scale level if the data is to be played backwards or is to loop back on it self.

Examples in the use of the SIMDIS_ScaleLevel generic data command can be found in the following distributed ASI file:

$(SIMDIS_DIR)/demos/SIMDIS/import/sampleGenericData2.asi

Track Color

```
# Sets the track's history color to it's original value
GenericData 602 "SIMDIS_TrackColor" "-1" "10:01:05.000"
GenericData 602 "SIMDIS_TrackColor" "-1" "10:04:09.900"
# Set the track history color to red (0xAABBGGRR) in hex
GenericData 602 "SIMDIS_TrackColor" "0xFF0000FF" "10:04:10.000"
GenericData 602 "SIMDIS_TrackColor" "0xFF0000FF" "10:07:09.900"
# Sets the track's history color to it's original value
GenericData 602 "SIMDIS_TrackColor" "-1" "10:07:10.000"
```

The format for the tag and data are as follows:

Where the keyword < SIMDIS_TrackColor > is the specific generic data tag. SIMDIS_TrackColor data contains only 1 required data field:

Track Color (char string) [**REQUIRED**]

This generic data tag is used to change the track history color for the referenced platform. Notice how the data is repeated in a time bracket. This must be done in order to preserve the correct track history color if the data is to be played backwards or is to loop back on it self. This generic data has been used to indicate a mode or status change for the associated platform.

Examples in the use of the SIMDIS_TrackColor generic data command can be found in the following distributed ASI files:

$(SIMDIS_DIR)/demos/SIMDIS/Users/SADM/simdis_pres_50001_0001.asi
$(SIMDIS_DIR)/demos/SIMDIS/Users/SADM/simdis_pres_60001_0001.asi

Vapor Trail

```
GenericData 602 "SIMDIS_VaporTrail 1" "initialRadius=1 ,fadeTime
=30,radiusExpansionRate=1,numRadiFromPreviousSmoke=2.5,metersBehindCu
rrentPostion=6,vaporTextureNames=vapor3.rgba:vapor4.rgba:vapor5.rgba:
vapor6.rgba" "10:01:05.000"
```

The format for the tag and data are as follows:

Where the keyword < SIMDIS_VaporTrail #> is the specific generic data tag. SIMDIS_VaporTrail data contains 6 required data fields:

initialRadius (meters)	[**REQUIRED**]
fadeTime (seconds)	[**REQUIRED**]
radiusExpansionRate (meters/seconds)	[**REQUIRED**]
numRadiFromPreviousSmoke (number of radius)	[**REQUIRED**]
metersBehindCurrentPostion (meters)	[**REQUIRED**]
vaporTextureNames (filenames)	[**REQUIRED**]

This generic data tag is used to display a vapor trail along the track history of the referenced platform. A series of "vapor items" are drawn along the platform's track history. The vapor items can expand and gradually fade out over time, essentially mimicking plane or rocket's vapor trail.

Examples in the use of the SIMDIS_VaporTrail generic data command can be found in the following distributed ASI files:

$(SIMDIS_DIR)/demos/SIMDIS/Examples/vaporTrail/vaporTrail.asi
$(SIMDIS_DIR)/demos/SIMDIS/import/sampleGenericData2.asi

Cylinder

```
GenericData 602 "SIMDIS_Cylinder 1"
"x=2.5,y=0,z=0,px=1,py=0,pz=0,l=4,radiusNear=.2,radiusFar=0,colorNear
=0xFF0000FF,colorFar=0xFFFFFFFF" "10:01:05.000"
```

The format for the tag and data are as follows:

Where the keyword < SIMDIS_Cylinder #> is the specific generic data tag. SIMDIS_Cylinder data contains 11 required data fields:

x (meters)	[**REQUIRED**]
y (meters)	[**REQUIRED**]
z (meters)	[**REQUIRED**]
px (pointing vector x component)	[**REQUIRED**]
py (pointing vector y component)	[**REQUIRED**]

pz (pointing vector z component)	[**REQUIRED**]
l (length in meters)	[**REQUIRED**]
radiusNear (meters)	[**REQUIRED**]
radiusFar (meters)	[**REQUIRED**]
colorNear (char string)	[**REQUIRED**]
colorFar (char string)	[**REQUIRED**]

This generic data tag is used to display a cylinder near the referenced platform. The cylinders have been used to display rocket burns. The Cylinder generic data ignores any specified duration value. SIMDIS interpolates the cylinder values when the SIMDIS time is in between two cylinder generic data points. To turn off the display of a cylinder, set the cylinder length value to zero.

Examples in the use of the SIMDIS_Cylinder generic data command can be found in the following distributed ASI files:

$(SIMDIS_DIR)/demos/SIMDIS/Examples/cylinder/cylinder.asi
$(SIMDIS_DIR)/demos/SIMDIS/import/sampleGenericData2.asi

Scale XYZ

```
GenericData 602 "SIMDIS_ScaleXYZ" "1000.0`1000.0`1000.0"
"10:01:05.000"
```

The format for the tag and data are as follows:

Where the keyword < SIMDIS_ScaleXYZ > is the specific generic data tag. SIMDIS_ScaleXYZ data contains 3 required data fields:

x (scale factor)	[**REQUIRED**]
y (scale factor)	[**REQUIRED**]
z (scale factor)	[**REQUIRED**]

This generic data tag is used to scale the referenced platform.

Examples in the use of the SIMDIS_ScaleXYZ generic data command can be found in the following distributed ASI files:

$(SIMDIS_DIR)/demos/SIMDIS/
Examples/scaleXYZAndAlphaVolume/scaleXYZAndAlphaVolume.asi
$(SIMDIS_DIR)/demos/SIMDIS/import/errorEllipse.asi

Alpha Volume

```
GenericData 602 "SIMDIS_AlphaVolume" "1" "10:01:05.000"
```

The format for the tag and data are as follows:

Where the keyword < SIMDIS_AlphaVolume > is the specific generic data tag. SIMDIS_AlphaVolume data contains 1 required data fields:

on/off (values are 1 and 0 respectively)	[**REQUIRED**]

This generic data tag has been used in conjunction with < SIMDIS_OverrideColor > generic data. If the < SIMDIS_OverrideColor > keyword is used, and if the color specified is semi-transparent, then you may want to add a < SIMDIS_AlphaVolume > line as well. Such a line will cause the platform's icon to be rendered in a manner similar to SIMDIS beams.

Examples in the use of the SIMDIS_AlphaVolume generic data command can be found in the following distributed ASI files:

$(SIMDIS_DIR)/demos/SIMDIS/
Examples/scaleXYZAndAlphaVolume/scaleXYZAndAlphaVolume.asi
$(SIMDIS_DIR)/demos/SIMDIS/import/errorEllipse.asi

Animated Lines

```
GenericData 602 "SIMDIS_AnimatedLine 1" "entityid=101 ,stipple
=0xFF00:0x00FF,color=0xFF14FF0E:0xFFFF4700,width=3.0,shiftsPerSecond=
60.0" "10:01:05.000"
```

The format for the tag and data are as follows:

Where the keyword < SIMDIS_AnimatedLine #> is the specific generic data tag. SIMDIS_AnimatedLine data contains 6 data fields:

entityid (Unique ID of another platform)	[**REQUIRED**]
stipple (two line stipple patterns separated by a :)	[**OPTIONAL**]
color (two color values separated by a :)	[**OPTIONAL**]
width (integer, line width)	[**OPTIONAL**]
shiftsPerSecond (float, animation speed and direction)	[**OPTIONAL**]
numSegments (integer, number of arc segements)	[**OPTIONAL**]

This generic data tag is used to display animated lines from one platform to another. This has been used to visualize communication between platforms.

Examples in the use of the SIMDIS_AnimatedLine generic data command can be found in the following distributed ASI files:

$(SIMDIS_DIR)/demos/SIMDIS/Examples/animatedLine/animatedLine.asi
$(SIMDIS_DIR)/demos/SIMDIS/import/animatedLine.asi

Draw Mode

```
GenericData 602 "SIMDIS_DrawMode" "solid" "10:01:05.000"
GenericData 602 "SIMDIS_DrawMode" "solid" "10:03:09.900"
GenericData 602 "SIMDIS_DrawMode" "point" "10:03:10.000"
GenericData 602 "SIMDIS_DrawMode" "point" "10:06:09.900"
GenericData 602 "SIMDIS_DrawMode" "wire"  "10:06:10.000"
```

The format for the tag and data are as follows:

Where the keyword < SIMDIS_DrawMode > is the specific generic data tag. SIMDIS_DrawMode data contains only 1 required data field:

Draw Mode (char string) [**REQUIRED**]

This generic data tag is used to change the draw mode for the referenced platform. Draw modes can range from solid, to wire frame and to a series of points. Notice how the data is repeated in a time bracket. This must be done in order to preserve the correct draw mode if the data is to be played backwards or is to loop back on it self. This command is typically used to indicate a particular state or mode such as track quality.

Examples in the use of the SIMDIS_DrawMode generic data command can be found in the following distributed ASI file:

$(SIMDIS_DIR)/demos/SIMDIS/import/sampleGenericData2.asi

A.3.7 True Type Fonts

When specifying a font in the ASI file format, both the font filename and size are required. Most integer values are valid, below are some examples:

```
// Courier New Bold font, with a point size of 15
"courb.ttf 15"

// Times New Roman Italic font, with a point size of 32
"timesi.ttf 32"
```

The following True Type Fonts and Font Families are available in SIMDIS.

Andale Mono, Version 2.0	
andalemo.ttf	Andale Mono
Arial, Version 2.76	
arial.ttf	Arial
arialbd.ttf	Arial Bold
arialbi.ttf	Arial Bold Italic
ariali.ttf	Arial Italic
Arial Black, Version 2.35	
ariblk.ttf	Arial Black
Comic Sans MS, Version 2.10	
comic.ttf	Comic Sans MS
comicbd.ttf	Comic Sans MS Bold
Courier New, Version 2.76	
cour.ttf	Courier New
courbd.ttf	Courier New Bold
courbi.ttf	Courier New Bold Italic
couri.ttf	Courier New Italic
Georgia, Version 2.05	
georgia.ttf	Georgia
georgiab.ttf	Georgia Bold
georgiai.ttf	Georgia Italic
georgiaz.ttf	Georgia Bold Italic

Impact, Version 2.35	
impact.ttf	Impact
Arial, Version 2.76	
times.ttf	Arial
timesbd.ttf	Arial Bold
timesbi.ttf	Arial Bold Italic
timesi.ttf	Arial Italic
Trebuchet MS, Version 1.15	
trebuc.ttf	Trebuchet MS
trebucbd.ttf	Trebuchet MS Bold
trebucbi.ttf	Trebuchet MS Bold Italic
trebucit.ttf	Trebuchet MS Italic
Verdana, Version 2.35	
verdana.ttf	Verdana
verdanab.ttf	Verdana Bold
verdanai.ttf	Verdana Italic
verdanaz.ttf	Verdana Bold Italic
Webdings	
webdings.ttf	Webdings

A.4 Version History

ASI Version 16

- Added the Platform keyword **PlatformOriOffset** to specify a rotation offset relative to a platform's body coordinates.

- Added the Gate keyword **BodyGateData** to allow azimuth and elevation offset values to be used with the body orientation of a host platform when drawing a gate.

ASI Version 15

- Added the Scenario keyword **RefLLA** to specify the reference latitude, longitude and altitude. Deprecated the scenario keywords **RefLat**, **RefLon** and **RefAlt**.

- Added support for two new coordinate systems, Earth Centered Inertial (ECI) and Generic Tangent Plane (GTP).

- Added "TRUE" as an argument accepted by the **MagneticVariance** keyword.

- Added the Scenario keyword **VerticalDatum** to specify the zero surface to which elevations or heights are referenced.

- Added the Scenario keyword **TangentPlaneOffset** to specify the x and y translation positions and a rotation angle to offset the Generic Tangent Plane (GTP) coordinate system.

- Added the Scenario keyword **ReferenceTimeECI** to specify the time at which the Earth Centered Inertial reference frame is defined.

- Replaced the Platform keyword **PlatformInterpolateOri** with **PlatformInterpolate** as interpolation is now controlled for all data, not just orientation. If the **PlatformInterpolateOri** keyword is encountered in older ASI files it will now behave as if it were the newer **PlatformInterpolate** keyword.

- Replaced the Beam keyword **BeamInterpolatePos** with **BeamInterpolate** to be consistent with the Platform version. If the **BeamInterpolatePos** keyword is encountered in older ASI files, it will behave as if it were the **BeamInterpolate** version.

- Replaced the Gate keyword **GateInterpolatePos** with **GateInterpolate** to be consistent with the Platform and Beam versions. If the **GateInterpolatePos**

keyword is encountered in older ASI files, it will behave as if it were the **GateInterpolate** version.

- Added the Platform keyword **PlatformUsesQuaternion** to indicate the use of a quaternion for specifying orientation instead of yaw, pitch and roll or Euler angles.

- Added support for platform independent coordinate systems which allows more than one system to be used in an ASI file. As such the following Platform keywords were added: **PlatformRefLLA**, **PlatformCoordSystem**, **PlatformMagneticVariance**, **PlatformVerticalDatum**, **PlatformReferenceTimeECI**, and **PlatformTangentPlaneOffset**. These keywords mimic their Scenario equivalents with the exception that they are applied at the Platform level.

- Added the Beam keyword **AntennaPolarity** to specify a beam's polarization.

- Added support for two new antenna pattern file formats, bilinear and the National Spectrum Managers Association (NSMA).

- Added support for a symmetrical RCS look up table. When the table setting is 2, the azimuth look up value is clamped between 0-180.

ASI Version 14

- Added the Beam keyword **BeamInterpolatePos** to control the interpolation of position between discrete data points.

- Added the Gate keyword **GateInterpolatePos** to control the interpolation of position between discrete data points.

ASI Version 13

- Added the Scenario keywords **ITConfigFile** and **RuleFile** to control the association of an imagery and terrain configuration file and a preference rules file to the data file.

ASI Version 12

- Added the Platform keyword **PlatformInterpolateOri** to control the interpolation of orientation between discrete data points.

ASI Version 11

- Added the generic data keyword **SIMDIS_DrawMode**

ASI Version 10

- Added the **OriginalID** keyword to Platforms, Beams and Gates.
- Added the **RefAlt** keyword to scenario initialization

ASI Version 9

- Changed time format strings; removed the weekday restriction.
- Added support for category data to Platforms, Beams and Gates.
- Changed Platform keywords **PlatformType** and **PlatformFHN** to become macros for creating Platform level category data.
- Added the Beam keyword **BeamMissileOffset** for automatically offsetting a missile seeker beam to the nose of the missile.

ASI Version 8

- Added the Beam keyword **BeamTargetIDCmd** as a replacement for the **BeamTargetCmd** keyword. **BeamTargetCmd** is now deprecated.
- Changed **SIMDIS_AnimatedLine** generic data to use an entity ID instead of a name.

ASI Version 7

- To avoid frequent confusion, generic data was changed to be a “property” of entities or of the scenario, rather than an object by itself.
- Deprecated: **GenericObjID**, **EventObjID**, **GenericObjRefID**, and **EventObjRefID**.

ASI Version 6

- Added the Scenario keyword **MagneticVariance** to specify the magnetic variance (declination) between magnetic north and true north.

- The Platform keyword **PlatformRCS** was added to handle the association of RCS files to a platform.

- Added the Platform keyword **PlatformAttachedGOG** to provide the ability to save GOG files attached to platforms.

- The Beam keyword **AREPSPattern** was added to handle the association of AREPS ASCII output text files to a beam.

- The Gate keyword **GateType** was added to indicate the type gate to be drawn.